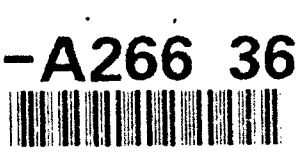


TESTING AND EVALUATING C^{3}I SYSTEMS THAT EMPLOY AI

(CLIN 0001)

VOLUME 5: TESTER_C USER'S MANUAL

Jacob W. Ulvila

Decision Science Consortium, Inc.
1895 Preston White Drive, Suite 300
Reston, Virginia 22091

January 1991

Final Report
Period of Performance: 16 September 1988 - 15 September 1990
Contract Number: DAEA18-88-C-0028
PR&C Number: W61DD3-8057-0601
AAP Number: EPG 8048

Prepared for:
U.S. Army Electronic Proving Ground
ATTN: STEEP-ET-S (Mr. Robert J. Harder)
Fort Huachuca, Arizona 85613-7110

TECHNICAL REPORT 90-9

REPORT DOCUMENTATION PAGE

Form Approved OMB No 0704-0188

1a. REPORT SECURITY CLASSIFICATION: Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Distribution subject to the SBIR Rights Notice (Jun 1987)
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): 90-9	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: Decision Science Consortium, Inc. / 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION: US Army Electronic Proving Ground STEEP-ET-S
6c. ADDRESS (City, State, and ZIP Code): 1895 Preston White Drive, Suite 300 Reston, Virginia 22091	7b. ADDRESS (City, State, and ZIP Code): Ft. Huachuca, Arizona 85613-7110
8a. NAME OF FUNDING/SPONSORING ORGANIZATION / 8b OFFICE SYMBOL (If applicable): STEEP-ET-S	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: DAEA-18-88-C-0028
8c. ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO / WORK UNIT ACCESSION NO

11. TITLE (Include Security Classification): TESTING AND EVALUATING C^3I SYSTEMS THAT EMPLOY AI -- VOLUME 5: TESTER_C USER'S MANUAL

12. PERSONAL AUTHOR(S): Jacob W. Ulvila

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final Technical	FROM Sep 88 TO Sep 90	1991 January 31	62

16. SUPPLEMENTARY NOTATION: The views, opinions, and/or findings contained in this report are those of the authors and should not be construed as an official Department of the Army position, policy, or decision unless so designated by other documentation.

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number): Expert Systems, Testing, Knowledge-Based Systems, Artificial Intelligence, Multiattribute Utility

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This volume is a user's manual for TESTER_C, a prototype computer program that implements the multiattribute utility framework for testing and evaluating expert systems that is described in Volume 1: Handbook for Testing Expert Systems. This user's manual describes how to load the program, input data, and analyze results. It also provides a brief overview of multiattribute utility analysis.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL: Mr. Robert J. Harder	22b. TELEPHONE (Include Area Code): (602) 538-2099 / 22c. OFFICE SYMBOL: STEEP-ET-S

DD Form 1473, JUN 86 — Previous editions are obsolete. — SECURITY CLASSIFICATION OF THIS PAGE: UNCLASSIFIED

CONTENTS

CONTENTS (Continued)

FIGURES

1.0 INTRODUCTION

This manual shows how to use the prototype software, TESTER_C. We proceed step-by-step through the menus and exercise the capabilities of the software. TESTER_C is a computer implementation of the multiattribute utility framework for testing and evaluating expert systems that is described in Volume 1, *Handbook for Testing Expert Systems*. This framework is summarized in Figure 1. The program is organized hierarchically by "menus" of alternative functions. The menu hierarchy is shown in Figure 2.

SYSTEM REQUIREMENTS

The program requires an IBM-PC/AT, IBM-XT or 100%-compatible computer, at least a single 1.2Mb diskette drive or a hard disk, 640K of internal RAM; and a parallel or serial I/O port for printing. A math co-processor is recommended, especially on slower machines. We recommend an EGA or VGA display for viewing utility curves, although CGA or Hercules can be used. If your computer has no graphics capability, you will not be able to view utility curves.

USING THE PROGRAM FOR THE FIRST TIME

We recommend that you first make backup copies of the diskette(s). We then recommend that you install TESTER_C on your hard disk, if you have one. Do this by creating a subdirectory on the disk called "AI2." Then copy all the files from the floppy to this subdirectory.

If your computer does not have a hard disk, proceed as follows. If your system has only a single drive, the computer will prompt you each time a diskette should be inserted. In the following instructions, you may use either upper- or lower-case. Proceed as follows:

(1) Place the DOS system diskette in Drive A (on the left), close the door, and then turn on electrical power to the computer. After a minute or so, you may be prompted to enter the date and time. Do so if requested, following each (and all later responses) with a carriage return. Separate the parts of the date with hyphens and the parts of the time with colons.

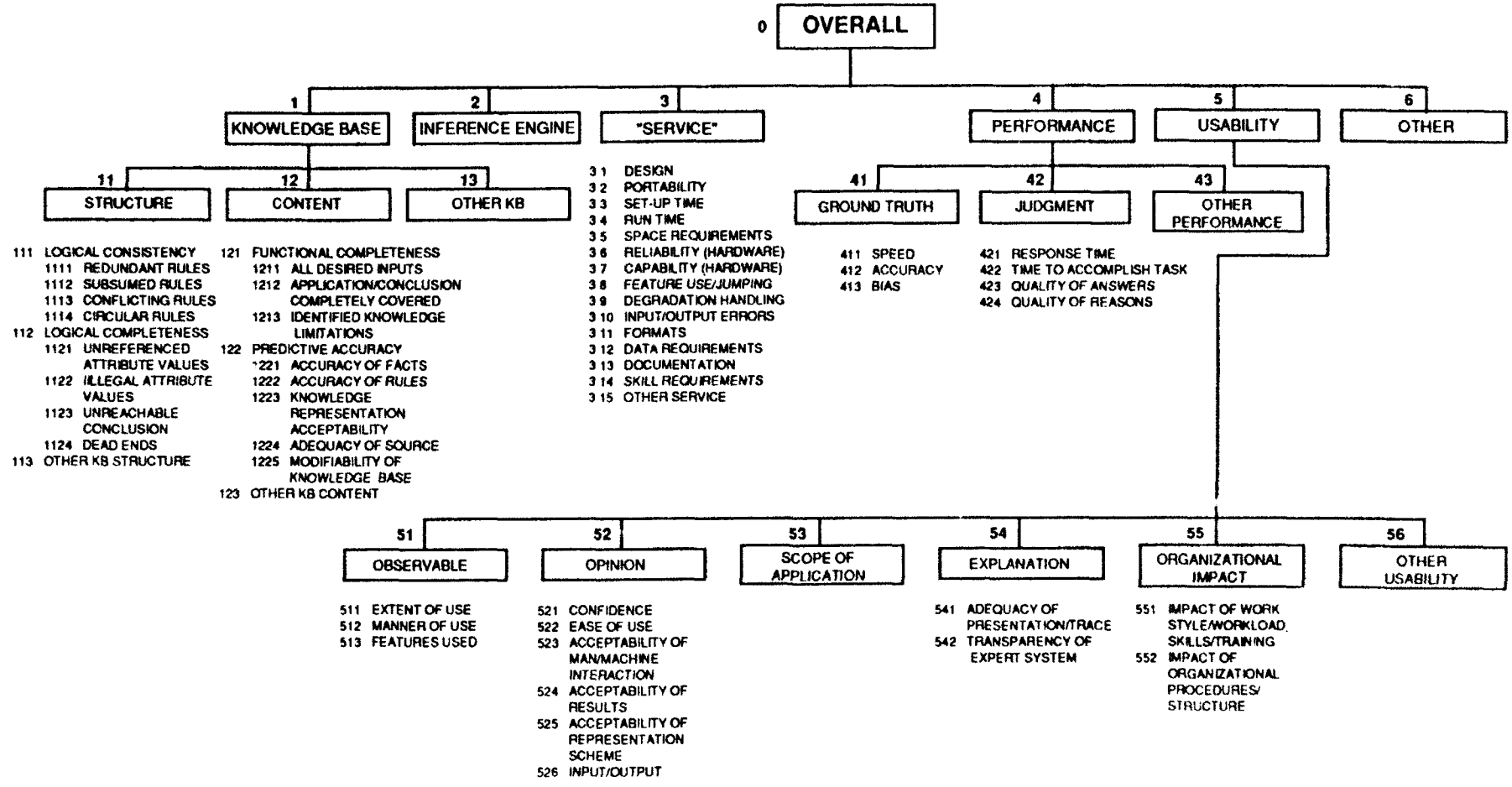


Figure 1: MAU Framework for Testing and Evaluating Expert Systems in TESTER_C

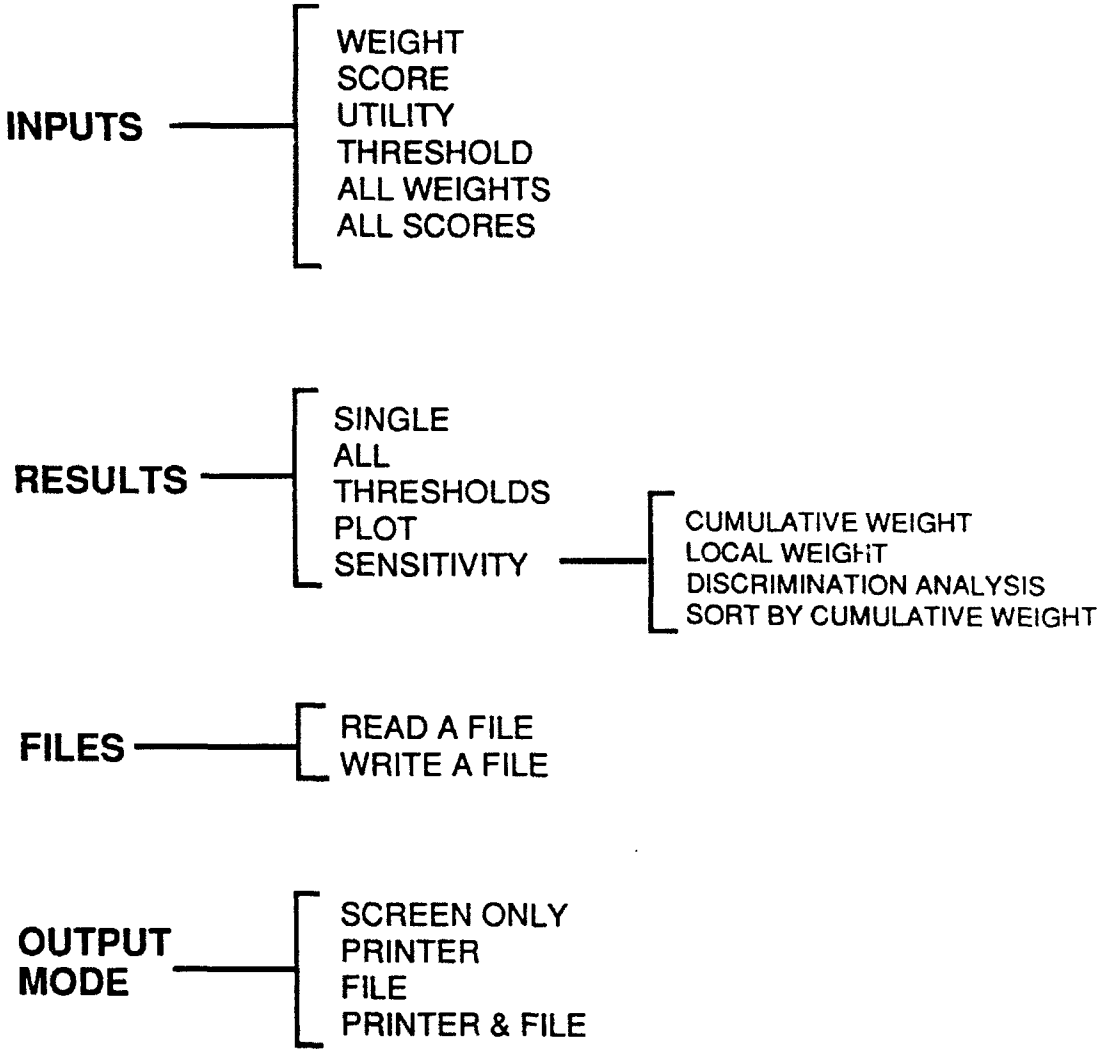


Figure 2: Menu Hierarchy

Following this, the DOS command prompt should appear:

```
A>
```

(2) Type the following command after the prompt:

```
DISKCOPY A: B:
```

A message will appear. Place the program diskette in the left (A:) drive and a blank diskette in the right (B:) drive. Press the return key. A message will appear when the copying process is complete—about a minute.

(3) Remove the right-hand diskette, label it as a file copy, and put it in a safe place. This provides a backup in case the original diskette is damaged or wears out.

(4) To use the program, it is necessary also to prepare diskettes to be used to store your models and data. If you do not have empty formatted diskettes handy, use the following procedure to produce one:

(i) Place the DOS system disk in drive A: and a blank diskette in drive B:. Type:

```
Format B:
```

(ii) When the formatting process ends, label the diskette as a "data diskette."

(5) To run the program, with the TESTER_C diskette in the default drive, type:

```
TESTER_C
```

SUGGESTIONS

In order to minimize the chance of lost data and wasted time, we recommend the following:

- Do not use the program diskette to hold data files; use a separate data diskette. You should place a write-protect tab on the program diskette. (You may also keep data files on your hard disk.)

- Keep track of how much space is available on your data diskettes, especially if you have several data files and are building models utilizing the "rationale" feature. To check the availability of space on a data diskette, use the following procedure. After

terminating the run by selecting TERMINATE PROGRAM, place the DOS or program diskette in the left-hand drive; press any key; place the data diskette in the right-hand drive; and type the directory command (DIR B:) after the system prompt (A>). The screen will show the names of files on the data diskette, the space used by each (in bytes), and the total amount of unused space on the diskette ("bytes free").

- Make a back-up copy of key data diskettes on a regular basis. To do this, you can use the DISKCOPY command illustrated above.

- To copy a model named MOD1 from a data diskette in the left drive to another in the right, type

```
A>COPY ?MOD1.ASF B:
```

DATA RIGHTS

TESTER_C is furnished to the U.S. Government with SBIR rights. The full SBIR Rights Notice is given in Appendix D.

2.0 USING THE PROGRAM

Start the program by typing "TESTER_C" to the DOS prompt. The program will ask the user where the data files are located as follows:

Indicate the disk drive where data files are located:			
Drive A	Drive B	Drive C	Drive D

Use the arrow keys to select the desired drive, hit the ENTER key (↵), and the following menu (later referred to as the main menu) will appear:

End	Inputs	Results	Files	Output Mode

END - used to terminate this section of the module.

INPUTS - used to change model inputs.

RESULTS - used to view the results of running the model.

FILES - used to read or write files.

OUTPUT MODE - used to select the destination for output.

Each of these selections is explained in more detail below.

LOADING THE MODEL

To load the model, it is necessary to access its file by selecting FILES from the following menu:

End	Inputs	Results	*Files*	Output Mode

After this selection, the following menu appears:

End	*Read a file*	Write a file

File to be read

-none-
SAMPLE
ZERO

Move with the Home, ↑↓, End keys and press Enter (or ESC to exit).

Select SAMPLE to load the model illustrated here. The file ZERO contains a model with the evaluation hierarchy, but zeros entered for all weights and scores.

SELECTING OUTPUT LOCATION

As the user begins, one of the first steps is to select the destination for the output by selecting OUTPUT MODE from the previous menu. The following appears:

Select the destination for output:			
Screen only	Printer	File	Printer + file

Under any selected option, the output appears on the screen. Possible choices are:

SCREEN ONLY - output appears at the screen only.

PRINTER - output appears on the screen and is sent to the printer.

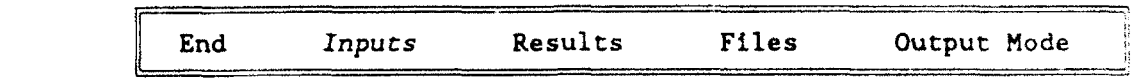
FILE - output appears on the screen and is sent to a data file named TESTER.PRN. This file is overwritten each time it is used (to save storage space). If you want to save several data files, exit the program and save the file under another name. The data file may be used later with a word processor to include TESTER_C output in a report.

PRINTER + FILE - output appears on the screen, at the printer, and in a data file.

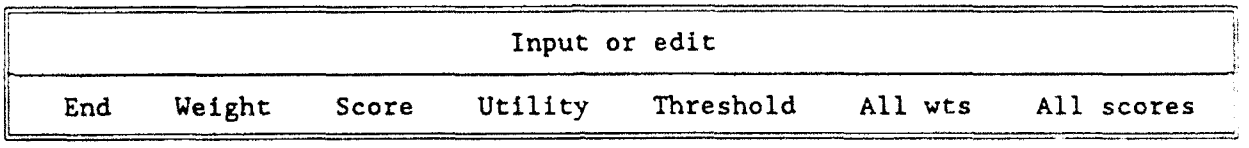
After a selection is made, hit the ESC (escape key) to return to the previous menu.

3.0 DATA INPUT

From the main menu, the user can input data by selecting INPUTS:



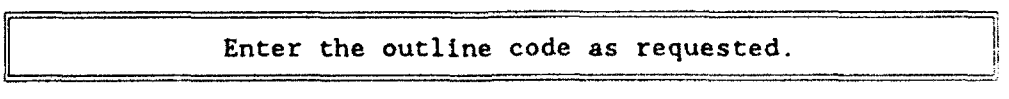
The following menu appears:



ALL SCORES and ALL WTS prompt the user with node numbers and require new entries for the entire structure. If WEIGHT or SCORE is selected, the user enters node numbers.

EDITING SCORES

To edit scores, the SCORE option is used. The following display appears:



Node for score editing:

If the user wanted to change the score for TES to 40 on node 5 2 1, he would enter the node number outline code as shown below (enter the node number with spaces between numbers):

Enter the outline code as requested.

Node for score editing: 5 2 1

When this is entered, the following appears:

Enter or edit the scores below. Use arrows to move between scores and 'ESC' to end. F2 = rationale edit & review.

5 2 1 Overall->Usability->Opinion->Confidenc

OPTIONS	OLD SCORES	NEW SCORES
tes	50.0	50.0
fal	0.0	0.0
mar	50.0	50.0
pas	50.0	50.0

The new score is typed as shown below. When all editing of scores is complete, hit ESC at the node prompt to return to the INPUT menu. Rationale, which is explained under Recording Rationale below may be reviewed by pressing the F2 key.

Enter or edit the scores below. Use arrows to move between scores and 'ESC' to end. F2 = rationale edit & review.

5 2 1 Overall->Usability->Opinion->Confidenc

OPTIONS	OLD SCORES	NEW SCORES
tes	50.0	40.0
fal	0.0	0.0
mar	50.0	50.0
pas	50.0	50.0

(For the purposes of this example, the score just edited is changed back to 50.)

EDITING WEIGHTS

To edit a weight, use the WEIGHT option on the INPUT menu. The following appears:

Enter the outline code as requested.

Node for weight editing:

To change the weights for knowledge-base structure and content to be equal, use the following sequence:

Enter the outline code as requested.

Node for weight editing: 1

The unedited display is shown as:

Edit the weights shown below. Use arrows to move between weights and 'ESC' to end. F2 = rationale edit & review, F9 = recompute.

1 Overall->Know_Base

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) Structure		0.40	36X	0X	45	50	0.090
2) Content		0.60	42	0	42	50	0.135
3) OtherKB	*	0.00	0	0	0	0	0.000
		----	----	----	----	----	----
COMBINED		1.00	40X	0X	43	50	0.224

After data entry, the display is:

Edit the weights shown below. Use arrows to move between weights and 'ESC' to end. F2 = rationale edit & review, F9 = recompute.

1 Overall->Know_Base

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) Structure		1	36X	0X	45	50	0.090
2) Content		1	42	0	42	50	0.135
3) OtherKB	*	0.00	0	0	0	0	0.000
		----	----	----	----	----	----
COMBINED		1.00	40X	0X	43	50	0.224

After recalculation (F9), the display is:

Edit the weights shown below. Use arrows to move between weights and 'ESC' to end. F2 = rationale edit & review, F9 = recompute.

1 Overall->Know_Base

ATTRIBUTE	WT	tes	fal	mar	pas	CUM. WT
1) Structure	0.50	36X	0X	45	50	0.112
2) Content	0.50	42	0	42	50	0.112
3) OtherKB	* 0.00	0	0	0	0	0.000
COMBINED	1.00	39X	0X	44	50	0.22

Note that when weights are edited and the model is recalculated, changes appear in the COMBINED line, the CUMWT column, the WT column, and in the SCORES columns if weights at lower levels have been edited.

(For the purposes of this example, weights are returned to their original values prior to the above editing.)

ENTERING THRESHOLDS

A criterion may be important enough that it would cause a system to fail by poor performance on that criterion alone, regardless of the performance on any other criteria. Such a minimum acceptable performance level is entered by selecting THRESHOLD. The following display appears asking the outline code of the node where the threshold is to be placed.

Enter the outline code as requested.

Node for threshold score:

Thresholds can be placed only at bottom-level nodes. For example, the following sets a threshold for node 1 1 1 4, circular rules.

Enter the outline code as requested.

Node for threshold score: 1 1 1 4

The following display appears:

Enter or edit the threshold below.
F2 = edit rationale.

1 1 1 4 Overall->Know_Base->Structure->Logic_Cons->CirculRuls

OLD SCORE	NEW SCORE
0	1

A value of 1 is set for this threshold as shown above. (Due to data structure limitations, thresholds cannot be set at 0 in this prototype software.) The user may define thresholds on as many nodes as he pleases. These thresholds are treated in the results as described below.

ENTERING UTILITY CURVES

In Appendix A, we describe how utility curves can be developed and scores for alternatives can be taken from those curves. Thus far, we have entered all scores directly into the model. It is also possible to build the curves into the model, to enter the natural values associated with an alternative, and have the utility scores automatically derived from the curves. We begin by selecting UTILITY from the INPUT menu. The following display appears asking for the outline code of the node where the utility curve is to be associated.

Enter the outline code as requested.

Node for utility curve:

Utility curves can be placed only at bottom-level nodes. For example, the following establishes a utility curve for node 3 4, run time.

Enter the outline code as requested.

Node for utility curve: 3 4

The following display appears:

Enter a label for the unit of measure for the utility function.

Unit of measure:

A six-character label can be assigned to each curve for the measure associated with the horizontal axis. In this case, the label is secs (seconds).

Enter a label for the unit of measure for the utility function.

Unit of measure: secs

The following display appears next:

Value	Utility
0	0
0	0
0	0
0	0
0	0
0	0
0	0
0	0
0	0
0	0

> Enter values in secs in the left column and the corresponding utilities in the right column. Order does not matter and blank rows are ignored. Press F7 to print, F8 to view utility curve, F2 to see rationale.

The user "defines" a utility curve by designating x-axis measure values along with associated y-axis utility values. Up to 10 points can be specified. The size of the utility value is limited by the 4 spaces allowed for entry. For example, if we associate a utility of 100 with a run time of 0 to 10 seconds, a utility of 50 with a run time of 20 seconds, a utility of 25 with a run time of 40 seconds, and a utility of 0 with a run time of 60 seconds, the entry is as follows: (Note: use the arrow (↑↓) keys to move back and forth among entries.)

Value	Utility
0	100
10	100
20	50
40	25
60	0
0	0
0	0
0	0
0	0
0	0

> Enter values in secs in the left column and the corresponding utilities in the right column. Order does not matter and blank rows are ignored. Press F7 to print, F8 to view utility curve, F2 to see rationale.

Before hitting ENTER, function key F8 can be used to view the curve. In constructing the utility curve, the software assumes piecewise, linear relationships between each successive pair of specified points.

After viewing the display, hit ESC to continue, and go into INPUT SCORES to enter the alternatives. In SAMPLE, the following is entered. Changes are made in the "New Scores" column.

Edit the scores shown below. Use arrows to move between scores and 'ESC' to end. F2 = rationale edit & review, F9 = calculate utility.

3 4 Overall->Service->RunTime

OPTIONS	OLD SCORES	NEW SCORES	OLD UTILITY	NEW UTILITY	‖Value units: secs ‖Values will be converted ‖to utilities. ‖Utilities are defined ‖for values in the range ‖0 to 60
tes	16.0	16	70.0	70	
fal	60.0	60	0.0	0	
mar	30.0	30	37.5	37.5	
pas	20.0	20	50.0	50	

Note that the display includes the label of the entered scores (secs) and the range of the values as determined from the specification of the utility curve. If the user tries to enter a score outside the defined range of the curve, a utility will be assigned equal to the closest defined point on the utility curve.

If 3 is selected from the RESULTS menu for SINGLE NODE, the following appears:

3 Overall->Service

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) Design	*	0.13	50	0	50	50	0.010
2) Portabilty	*	0.00	0	0	0	50	0.000
3) SetUpTime	*	0.00	20	0	25	50	0.000
4) RunTime	u	0.13	70	0	38	50	0.010
5) SpaceReqs	*	0.06	50	0	25	50	0.005
6) Reliabilty	*	0.06	50	0	50	50	0.005
7) Capability	*	0.06	50	0	50	50	0.005
8) FeatureUse	*	0.06	50	0	30	50	0.005
9) DegrdHandl	*	0.13	25	0	50	50	0.010
10) I_O_Errors	*	0.13	50	0	50	50	0.010
11) Formats	*	0.06	50	0	50	50	0.005
12) DataReqs	*	0.06	50	0	50	50	0.005
13) Documentn	*	0.06	20	0	25	50	0.005
14) SkillReqs	*	0.06	50	0	50	50	0.005
COMBINED		1.00	48	0	44	50	0.081

Since the scores shown for node 3 4 are those generated by a utility curve, a "u" appears next to RunTime, while a "*" appears next to the other attributes. Both symbols indicate that the node is a bottom-level node for which scores or values that generate scores are entered directly. The "u" indicates that a utility curve exists for this node.

4.0 ANALYSIS OF RESULTS

DISPLAY RESULTS

Alternatives are compared using a matrix format for displaying results. To display results, select RESULTS from the main menu.

End	Inputs	*Results*	Files	Output Mode

The following appears:

End	Single	All	Thresholds	Plot	Sensitivity

To view results for a single node, such as node 0, select SINGLE NODE, enter the outline code at the prompt, and hit ENTER:

Enter the outline code as requested.

Node for results display: 0

Hit any key to proceed...

0 Overall

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) Know_Base		0.22	40X	0X	43	50	0.224
2) InferncEng	*	0.02	50	0	50	50	0.020
3) Service		0.08	48	0	44	50	0.081
4) Perfrmanc		0.22	65	0	40	50	0.224
5) Usability		0.45	55	0	39	50	0.450
6) OtherTech	*	0.00	0	0	0	0	0.000
COMBINED		1.00	53X	0X	41	50	1.000

This matrix shows the value of each of the four systems against each of the six top-level attributes. The COMBINED row is the weighted average of these values. The value for inference engine was entered (indicated by the *) and the other values were calculated from lower-level nodes. Note that tes is shown with an X beside its value on *Know_Base* and COMBINED. This indicates that, although its weighted score is the highest shown, it fails at least one knowledge base attribute that has a threshold.

A faster check against thresholds is given by selecting THRESHOLDS, which produces the following screen:

System to check against thresholds

-none-
tes
fal
mar
pas

If tes is selected, then the following appears, indicating that it fails the knowledge base standard that prohibits circular rules. (If more than one threshold were violated, they would all be listed.)

"tes" fails the following thresholds: <Hit any key to exit>

1 1 1 4 >Know_Base>Structure>Logic_Cons>CirculRuls

To view all matrices in the model, select ALL NODES. A complete set of matrices is shown below.

---

0 Overall

ATTRIBUTE	WT	tes	fal	mar	pas	CUM. WT
1) Know_Base	0.22	40X	0X	43	50	0.224
2) InferncEng *	0.02	50	0	50	50	0.020
3) Service	0.08	48	0	44	50	0.081
4) Perfrmanc	0.22	65	0	40	50	0.224
5) Usability	0.45	55	0	39	50	0.450
6) OtherTech *	0.00	0	0	0	0	0.000
	----	----	----	----	----	----
COMBINED	1.00	53X	0X	41	50	1.000

1 Overall->Know_Base

ATTRIBUTE	WT	tes	fal	mar	pas	CUM. WT
1) Structure	0.40	36X	0X	45	50	0.090
2) Content	0.60	42	0	42	50	0.135
3) OtherKB *	0.00	0	0	0	0	0.000
	----	----	----	----	----	----
COMBINED	1.00	40X	0X	43	50	0.224

1 1 Overall->Know_Base->Structure

ATTRIBUTE	WT	tes	fal	mar	pas	CUM. WT
1) Logic_Cons	0.67	29X	0X	43	50	0.060
2) Logic_Comp	0.33	50	0	50	50	0.030
3) OtherStruc *	0.00	0	0	0	0	0.000
	----	----	----	----	----	----
COMBINED	1.00	36X	0X	45	50	0.090

1 1 1 Overall->Know_Base->Structure->Logic_Cons

ATTRIBUTE	WT	tes	fal	mar	pas	CUM. WT
1) RedunRules *	0.18	40	0	30	50	0.011
2) SubsumRuls *	0.18	25	0	30	50	0.011
3) ConflctRul *	0.36	50	0	50	50	0.021
4) CirculRuls *	0.29	0X	0X	50	50	0.017
	----	----	----	----	----	----
COMBINED	1.00	29X	0X	43	50	0.060

1 1 2 Overall->Know_Base->Structure->Logic_Comp

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) Unreferenc	*	0.29	50	0	50	50	0.009
2) Ill_Attrib	*	0.29	50	0	50	50	0.009
3) UnreachCon	*	0.29	50	0	50	50	0.009
4) Dead_Ends	*	0.14	50	0	50	50	0.004
COMBINED		1.01	50	0	50	50	0.030

1 2 Overall->Know_Base->Content

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) Func_Comp		0.50	38	0	38	50	0.067
2) Pred_Accy		0.50	46	0	46	50	0.067
3) OtherCont	*	0.00	0	0	0	0	0.000
COMBINED		1.00	42	0	42	50	0.135

1 2 1 Overall->Know_Base->Content->Func_Comp

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) AllDesInpt	*	0.25	50	0	50	50	0.017
2) CompCover	*	0.50	50	0	50	50	0.034
3) IdKnowLimt	*	0.25	0	0	0	50	0.017
COMBINED		1.00	38	0	38	50	0.067

1 2 2 Overall->Know_Base->Content->Pred_Accy

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) AccFacts	*	0.30	50	0	50	50	0.020
2) AccRules	*	0.30	50	0	50	50	0.020
3) KnowRepAc	*	0.15	50	0	50	50	0.010
4) AdeqSrce	*	0.09	50	0	50	50	0.006
5) Modif_KB	*	0.15	25	0	25	50	0.010
COMBINED		1.00	46	0	46	50	0.067

3 Overall->Service

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) Design	*	0.13	50	0	50	50	0.010
2) Portabilty	*	0.00	0	0	0	50	0.000
3) SetUpTime	*	0.00	20	0	25	50	0.000
4) RunTime	u	0.13	70	0	38	50	0.010
5) SpaceReqs	*	0.06	50	0	25	50	0.005
6) Reliabilty	*	0.06	50	0	50	50	0.005
7) Capability	*	0.06	50	0	50	50	0.005
8) FeatureUse	*	0.06	50	0	30	50	0.005
9) DegrdHandl	*	0.13	25	0	50	50	0.010
10) I_O_Errors	*	0.13	50	0	50	50	0.010
11) Formats	*	0.06	50	0	50	50	0.005
12) DataReqs	*	0.06	50	0	50	50	0.005
13) Documentn	*	0.06	20	0	25	50	0.005
14) SkillReqs	*	0.06	50	0	50	50	0.005
COMBINED		1.00	48	0	44	50	0.081

4 Overall->Perfrmanc

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) GrndTruth		0.20	75	0	35	50	0.045
2) Judgment		0.80	62	0	41	50	0.180
3) OtherPerf	*	0.00	0	0	0	0	0.000
COMBINED		1.00	65	0	40	50	0.224

4 1 Overall->Perfrmanc->GrndTruth

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) Speed	*	0.00	20	0	25	50	0.000
2) Accuracy	*	0.67	75	0	40	50	0.030
3) Bias	*	0.33	75	0	25	50	0.015
COMBINED		1.00	75	0	35	50	0.045

4 2 Overall->Perfrmanc->Judgment

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) RsponsTime	*	0.37	75	0	25	50	0.067
2) TimetoTask	*	0.19	75	0	50	50	0.033
3) QualAnswrs	*	0.37	50	0	50	50	0.067
4) QualReasns	*	0.07	25	0	50	50	0.013
COMBINED		1.00	62	0	41	50	0.180

5 Overall->Usability

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) Observble		0.07	45	0	40	50	0.030
2) Opinion		0.33	53	0	40	50	0.150
3) ScopeofApp	*	0.27	75	0	40	50	0.120
4) Explanatn		0.17	38	0	50	50	0.075
5) OrgImpact		0.17	50	0	25	50	0.075
6) OtherUsbl	*	0.00	0	0	0	0	0.000
COMBINED		1.00	55	0	39	50	0.450

5 1 Overall->Usability->Observble

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) ExtentUse	*	0.40	50	0	50	50	0.012
2) MannerUse	*	0.20	25	0	50	50	0.006
3) FeaturUse	*	0.40	50	0	25	50	0.012
COMBINED		1.00	45	0	40	50	0.030

5 2 Overall->Usability->Opinion

ATTRIBUTE		WT	tes	fal	mar	pas	CUM. WT
1) Confidenc	*	0.20	50	0	50	50	0.030
2) EaseofUse	*	0.20	75	0	25	50	0.030
3) AccMMI	*	0.20	50	0	25	50	0.030
4) AccResults	*	0.20	50	0	50	50	0.030
5) AccRepSchm	*	0.10	50	0	50	50	0.015
6) Inp_Out	*	0.10	25	0	50	50	0.015
COMBINED		1.00	53	0	40	50	0.150

5 4 Overall->Usability->Explanatn

ATTRIBUTE	WT	tes	fal	mar	pas	CUM. WT
1) AdeqPresen *	0.50	50	0	50	50	0.037
2) TransparES *	0.50	25	0	50	50	0.037
COMBINED	1.00	38	0	50	50	0.075

5 5 Overall->Usability->OrgImpact

ATTRIBUTE	WT	tes	fal	mar	pas	CUM. WT
1) Workload *	0.33	50	0	25	50	0.025
2) Procedure *	0.67	50	0	25	50	0.050
COMBINED	1.00	50	0	25	50	0.075

PLOTTING RESULTS

One feature of the software is to allow the user to plot any criterion against any other and compare alternatives. For example, the user may want to plot performance on knowledge base against performance on usability. Select PLOT from the RESULTS menu and the following appears:

To define a plot, enter the data requested below.
Use arrow keys to move among items.

Outline code for x-axis:
Outline code for y-axis:

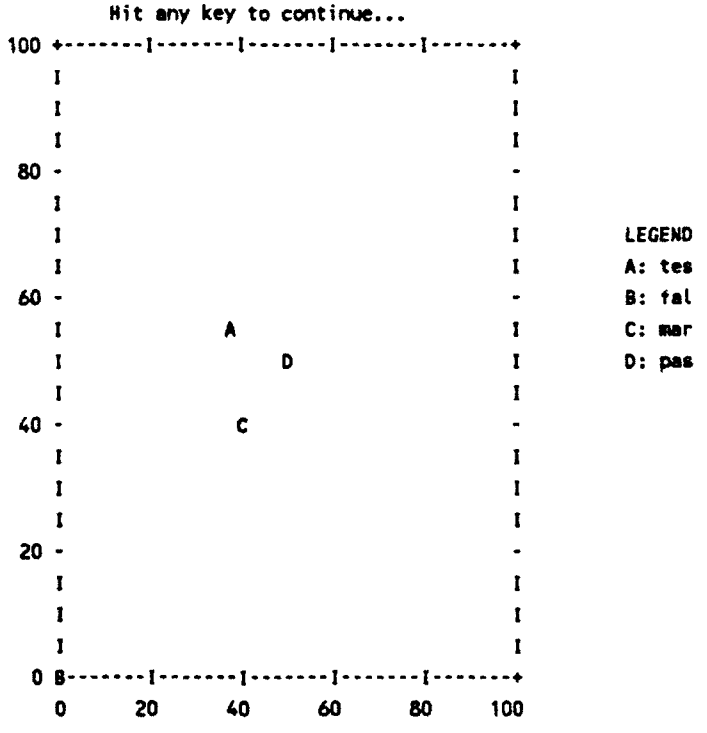
	Minimum	Maximum
x-axis	0	100
y-axis	0	100

The user specifies which outline code should be placed on each axis, and can change the scales if he desires. For example, we will plot node 1, knowledge base, on the x-axis and node 5, usability, on the y-axis. Each axis has utility values running from 0 to 100.

To define a plot, enter the data requested below.
Use arrow keys to move among items.

Outline code for x-axis: 1
Outline code for y-axis: 5

	Minimum	Maximum
x-axis	0	100
y-axis	0	100



The plot will appear on the screen. To get a printout of this plot, you must use Shift PrtSc (print screen). The legend block explains the key for identifying the alternatives. For example, the letter A represents the test

system, which scores approximately 40 on knowledge base and 55 on usability according to the plot.

By noticing that no value exceeds 60, we can get a finer level of detail by changing the scale limits on the axes to 0 and 60 rather than 0 to 100.

SENSITIVITY ANALYSIS

Several types of sensitivity analysis are offered. The user begins by selecting SENSITIVITY from the RESULTS menu:

End	Single	All	Thresholds	Plot	*Sensitivity*

The following menu appears:

End	Cum wt sensitivity	Discrimination analysis
	Local wt sensitivity	Sort by cum wt

CUM WT SENSITIVITY - This allows the user to vary the cumulative weight of any bottom-level node.

DISCRIMINATION ANALYSIS - This allows the user to do direct comparisons of alternatives, for example, the system under test could be compared attribute-by-attribute with a marginal system, with the passing system, or with the complete failure.

LOCAL WT SENSITIVITY - This allows the user to vary a weight at any node in the hierarchy.

SORT BY CUM WT - This allows the user to produce a list of bottom-level criteria in order of decreasing CUM WT.

Sensitivity on Local Weight

The user can trace the effects of varying a weight at any node by selecting LOCAL WT SENSITIVITY from the SENSITIVITY menu.

End	Cum wt sensitivity *Local wt sensitivity*	Discrimination analysis Sort by cum wt

---

The following display appears:

LOCAL WEIGHT SENSITIVITY

Outline code for weight to be varied:

Minimum weight: 0 Maximum weight: 1

Suppose the user wanted to examine the effects of changing the weight of node 1, knowledge base, which currently has a weight of .22. He can let it range from a low weight of 0 to a high weight of 1.0 by making the following entries:

LOCAL WEIGHT SENSITIVITY

Outline code for weight to be varied: 1

Minimum weight: 0 Maximum weight: 1

The software automatically divides this range into ten intervals, with each interval representing the weight for node 1. The weights for all other factors at the selected node are calculated by taking the remaining weight

after excluding the sensitivity node and allocating it in the same proportions as were the original weights. The software then calculates the overall evaluation for each set of weights. This is the equivalent of the COMBINED line for node 0.

LOCAL WEIGHT SENSITIVITY

Outline code for weight to be varied: 1

Minimum weight: 0 Maximum weight: 1

LOCAL WEIGHT SENSITIVITY
Outline code for weight to be varied: 1
From: 0.00 To: 1.00
Node name: Overall->Know_Base
CURRENT LOCAL WEIGHT = 0.224

WEIGHT	tes	fal	mar	pas
0.00	56*	0	40	50
0.10	55*	0	40	50
0.20	53*	0	40	50
0.30	51*	0	41	50
0.40	50*	0	41	50
0.50	48	0	41	50*
0.60	46	0	41	50*
0.70	44	0	42	50*
0.80	43	0	42	49*
0.90	41	0	42	49*
1.00	39	0	43	49*

The display provides the current weight for the factor as assigned in the model (i.e., .2244). The remainder of the table is read as follows: "if the weight of node 1 were the value in the WEIGHT column, the overall evaluations of the alternatives would be as shown to the right of the weight." Thus, if knowledge base had a weight of .30, tes would be evaluated at 51, fal at 0, mar at 41, and pas at 50. The asterisk indicates the alternative with the highest score in the row (this analysis ignores thresholds). We can see that the highest among alternatives changes from tes to pas when the weight exceeds .4 (the display rounds to two places, so both appear with 50s in this

range). To get a more precise look at where the change occurs, we can run the same sensitivity between .4 and .5 as follows:

LOCAL WEIGHT SENSITIVITY

Outline code for weight to be varied: 1

Minimum weight: .4 Maximum weight: .5

LOCAL WEIGHT SENSITIVITY
Outline code for weight to be varied: 1
From: 0.40 To: 0.50
Node name: Overall->Know_Base
CURRENT LOCAL WEIGHT = 0.224

WEIGHT	tes	fal	mar	pas
0.40	50*	0	41	50
0.41	49	0	41	50*
0.42	49	0	41	50*
0.43	49	0	41	50*
0.44	49	0	41	50*
0.45	49	0	41	50*
0.46	48	0	41	50*
0.47	48	0	41	50*
0.48	48	0	41	50*
0.49	48	0	41	50*
0.50	48	0	41	50*

Local weight sensitivity can be done at any node in the structure. The weights that have been entered into the node previously are unchanged after ending the sensitivity sequence.

Cumulative Weight Sensitivity

Occasionally, the user may want to determine the effect of making a more global change in the weights. He can do this by changing the CUMWT for a node rather than its local weight. For example, we can see above that the CUMWT for knowledge base content is .135. While it represents 60% of the weight under knowledge base, the CUMWT indicates that it accounts for approximately 13.5% of the entire model. The user can vary this CUMWT and view the results

in the same format as described above. The sequence would be as follows for varying CUMWT at node 1 2 between .1 and .5:

End	*Cum wt sensitivity* Local wt sensitivity	Discrimination analysis Sort by cum wt

---

CUMULATIVE WEIGHT SENSITIVITY

Outline code for weight to be varied: 1 2

Minimum weight: .1 Maximum weight: .5

CUMULATIVE WEIGHT SENSITIVITY
Outline code for weight to be varied: 1 2
From: 0.10 To: 0.50
Node name: Overall->Know_Base->Content
CURRENT CUM WT = 0.135

WEIGHT	tes	fal	mar	pas
0.10	53*	0	40	50
0.14	52*	0	40	50
0.18	52*	0	40	50
0.22	51*	0	40	50
0.26	51*	0	40	50
0.30	50*	0	40	50
0.34	50*	0	41	50
0.38	49	0	41	50*
0.42	49	0	41	50*
0.46	48	0	41	50*
0.50	48	0	41	50*

As before, all weight not assigned to the node on which sensitivity is being tested is reallocated in the original proportions.

Sort by Cumulative Weight

In order to determine which criteria are the major factors in the test evaluation, it is useful to list the bottom-level criteria in order. A list

of these criteria can be obtained by selecting SORT BY CUM WT from the SENSITIVITY menu as follows:

End	Cum wt sensitivity Local wt sensitivity	Discrimination analysis *Sort by cum wt*

ATTRIBUTE-BY-ATTRIBUTE SCORES IN ORDER OF CUMULATIVE WEIGHT ON EACH ATTRIBUTE

NODE	ATTRIBUTE	tes	fal	mar	pas	CUMWT	TOTAL
4 1 1	Speed	20	0	25	50	0.0000	0.0000
3 3	SetUpTime	20	0	25	50	0.0000	0.0000
6	OtherTech	0	0	0	0	0.0000	0.0000
5 6	OtherUsbl	0	0	0	0	0.0000	0.0000
3 2	Portabilty	0	0	0	50	0.0000	0.0000
1 3	OtherKB	0	0	0	0	0.0000	0.0000
1 2 3	OtherCont	0	0	0	0	0.0000	0.0000
4 3	OtherPerf	0	0	0	0	0.0000	0.0000
1 1 3	OtherStruc	0	0	0	0	0.0000	0.0000
3 15	OthService	0	0	0	0	0.0000	0.0000
1 1 2 4	Dead_Ends	50	0	50	50	0.0043	0.0043
3 12	DataReqs	50	0	50	50	0.0051	0.0093
3 13	Documentn	20	0	25	50	0.0051	0.0144
3 14	SkillReqs	50	0	50	50	0.0051	0.0195
3 11	Formats	50	0	50	50	0.0051	0.0246
3 8	FeatureUse	50	0	30	50	0.0051	0.0296
3 7	Capability	50	0	50	50	0.0051	0.0347
3 6	Reliabilty	50	0	50	50	0.0051	0.0398
3 5	SpaceReqs	50	0	25	50	0.0051	0.0449
5 1 2	MannerUse	25	0	50	50	0.0060	0.0509
1 2 2 4	AdeqSrce	50	0	50	50	0.0061	0.0570
1 1 2 1	Unreferenc	50	0	50	50	0.0086	0.0655
1 1 2 3	UnreachCon	50	0	50	50	0.0086	0.0741
1 1 2 2	Ill_Attrib	50	0	50	50	0.0086	0.0826
3 4	RunTime	16	60	30	20	0.0101	0.0928
3 1	Design	50	0	50	50	0.0101	0.1029
3 9	DegrdHandl	25	0	50	50	0.0101	0.1131
3 10	I_O_Errors	50	0	50	50	0.0101	0.1232
1 2 2 5	Modif_KB	25	0	25	50	0.0102	0.1334
1 2 2 3	KnowRepAc	50	0	50	50	0.0102	0.1436
1 1 1 2	SubsumRuls	25	0	30	50	0.0107	0.1543
1 1 1 1	RedunRules	40	0	30	50	0.0107	0.1650
5 1 3	FeaturUse	50	0	25	50	0.0120	0.1770
5 1 1	ExtentUse	50	0	50	50	0.0120	0.1890
4 2 4	QualReasns	25	0	50	50	0.0133	0.2023
4 1 3	Bias	75	0	25	50	0.0150	0.2172
5 2 5	AccRepSchm	50	0	50	50	0.0150	0.2322

5	2	6		Inp_Out	25	0	50	50 0.0150 0.2472
1	2	1	3	IdKnowLimt	0	0	0	50 0.0168 0.2641
1	2	1	1	AllDesInpt	50	0	50	50 0.0168 0.2809
1	1	1	4	CirculRuls	0	0	50	50 0.0171 0.2980
2				InferncEng	50	0	50	50 0.0200 0.3180
1	2	2	2	AccRules	50	0	50	50 0.0204 0.3385
1	2	2	1	AccFacts	50	0	50	50 0.0204 0.3589
1	1	1	3	ConflctRul	50	0	50	50 0.0214 0.3802
5	5	1		Workload	50	0	25	50 0.0250 0.4052
4	1	2		Accuracy	75	0	40	50 0.0299 0.4352
5	2	2		EaseofUse	75	0	25	50 0.0300 0.4651
5	2	3		AccMMI	50	0	25	50 0.0300 0.4951
5	2	4		AccResults	50	0	50	50 0.0300 0.5251
5	2	1		Confidenc	50	0	50	50 0.0300 0.5551
4	2	2		TimetoTask	75	0	50	50 0.0333 0.5884
1	2	1	2	CompCover	50	0	50	50 0.0337 0.6220
5	4	1		AdeqPresen	50	0	50	50 0.0375 0.6595
5	4	2		TransparES	25	0	50	50 0.0375 0.6970
5	5	2		Procedure	50	0	25	50 0.0500 0.7470
4	2	3		QualAnswrs	50	0	50	50 0.0665 0.8135
4	2	1		RsponsTime	75	0	25	50 0.0665 0.8800
5	3			ScopeofApp	75	0	40	50 0.1200 1.0000

Note that only bottom-level factors are shown, and the scores that appear are those that were entered into the model either directly or through utility curves. In this case, the eleven highest-weighted criteria account for over half of the entire evaluation (the list is in increasing order of weight).

Discrimination Analysis

It is often useful for the tester to compare directly the system under test with another system, such as the passing system, to highlight its strengths and weaknesses. This is done by selecting DISCRIMINATION ANALYSIS from the SENSITIVITY menu:

End	Cum wt sensitivity	*Discrimination analysis*
	Local wt sensitivity	Sort by cum wt

The following then appears:

Choose the first package for comparison.

tes
fal
mar
pas

To compare the system under test against the performance targets, select test, hit ENTER, select pas, and hit ENTER again. The sequence is as follows:

Choose the first package for comparison.

tes
fal
mar
pas

Choose a second, different package.

tes
fal
mar
pas

ATTRIBUTE-BY-ATTRIBUTE COMPARISON BETWEEN tes AND pas

NODE	LABEL	WEIGHTED DIFF	CUM WTD DIFF
5 4 2	TransparES	-0.9373	-0.9373
1 1 1 4	CirculRuls	-0.8550	-1.7923
1 2 1 3	IdKnowLimt	-0.8417	-2.6340
5 2 6	Inp_Out	-0.3749	-3.0089
4 2 4	QualReasns	-0.3325	-3.3415
1 1 1 2	SubsumRuls	-0.2672	-3.6087
1 2 2 5	Modif_KB	-0.2551	-3.8637
3 9	DegrdHandl	-0.2536	-4.1173
3 13	Documentn	-0.1522	-4.2695
5 1 2	MannerUse	-0.1500	-4.4195

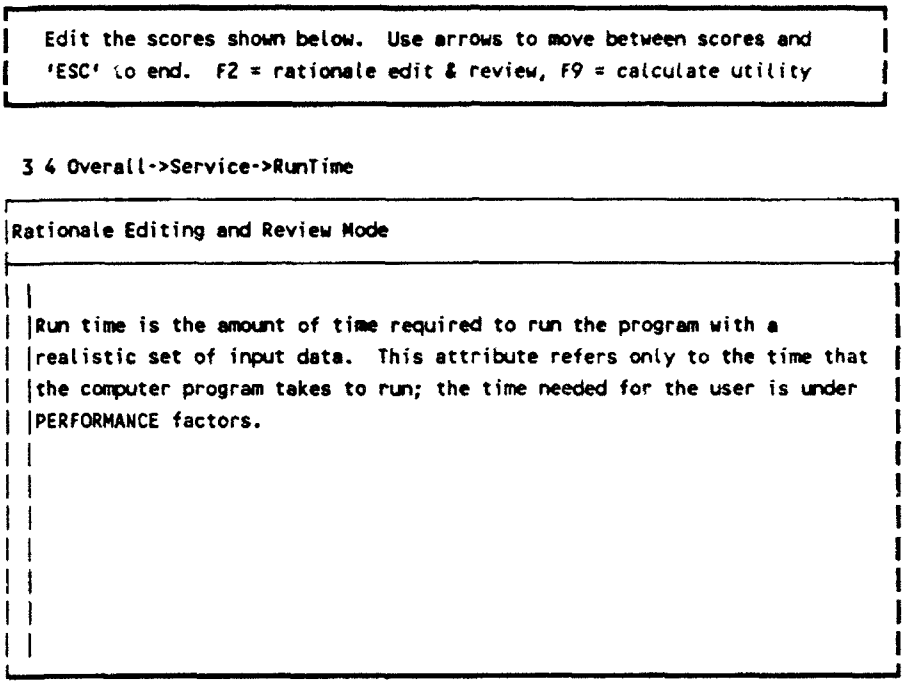
1 1 1 1	RedunRules	-0.1069	-4.5264
3 4	RunTime	-0.0406	-4.5669
1 2 2 1	AccFacts	0.0000	-4.3235
1 2 2 2	AccRules	0.0000	-4.3235
1 2 2 3	KnowRepAc	0.0000	-4.3235
1 2 2 4	AdeqSrce	0.0000	-4.3235
1 2 1 1	AllDesInpt	0.0000	-4.3235
1 2 3	OtherCont	0.0000	-4.3235
1 3	OtherKB	0.0000	-4.3235
2	InferncEng	0.0000	-4.3235
3 1	Design	0.0000	-4.3235
3 2	Portabilty	0.0000	-4.3235
3 3	SetUpTime	0.0000	-4.3235
1 2 1 2	CompCover	0.0000	-4.3235
3 5	SpaceReqs	0.0000	-4.3235
3 6	Reliabilty	0.0000	-4.3235
3 7	Capability	0.0000	-4.3235
3 8	FeatureUse	0.0000	-4.3235
1 1 3	OtherStruc	0.0000	-4.3235
3 10	I_O_Errors	0.0000	-4.3235
3 11	Formats	0.0000	-4.3235
3 12	DataReqs	0.0000	-4.3235
1 1 2 4	Dead_Ends	0.0000	-4.3235
3 14	SkillReqs	0.0000	-4.3235
3 15	OthService	0.0000	-4.3235
4 1 1	Speed	0.0000	-4.3235
1 1 1 3	ConflctRul	0.0000	-4.3235
5 4 1	AdeqPresen	0.0000	-4.3235
6	OtherTech	0.0000	-4.3235
5 6	OtherUsbl	0.0000	-4.3235
4 2 3	QualAnswrs	0.0000	-4.3235
1 1 2 3	UnreachCon	0.0000	-4.3235
4 3	OtherPerf	0.0000	-4.3235
5 1 1	ExtentUse	0.0000	-4.3235
1 1 2 2	Ill_Attrib	0.0000	-4.3235
5 1 3	FeaturUse	0.0000	-4.3235
5 2 1	Confidenc	0.0000	-4.3235
5 5 2	Procedure	0.0000	-4.3235
5 2 3	AccMMI	0.0000	-4.3235
5 2 4	AccResults	0.0000	-4.3235
5 2 5	AccRepSchm	0.0000	-4.3235
1 1 2 1	Unreferenc	0.0000	-4.3235
5 5 1	Workload	0.0000	-4.3235
4 1 3	Bias	0.3741	-3.9494
4 1 2	Accuracy	0.7482	-3.2012
5 2 2	EaseofUse	0.7498	-2.4514
4 2 2	TimetoTask	0.8313	-1.6201
4 2 1	RsponsTime	1.6626	0.0425
5 3	ScopeofApp	2.9993	3.0418

The column labeled WEIGHTED DIFF is calculated by taking the differences in scores between tes and pas on each criterion and multiplying the difference by the CUMWT for that criterion. The rightmost column keeps a ru ing total of CUMWT weighted differences as the display proceeds through the criteria. The criteria are shown in the order that starts with the criterion where tes most importantly falls below the standard and ends with the one where tes most importantly exceeds the standard. Any positive value means that tes exceeds the standard, a 0 value indicates that tes meets the standard, and a negative value indicates that tes is below the standard. The final value in the right column is the final difference in score between tes and pas in the overall evaluation (within roundoff). The net positive value indicates that, overall, the test system exceeds the standard (ignoring thresholds).

RECORDING RATIONALE

The software also allows rationale to be recorded. Rationale can be reviewed and revised by selecting RATIONALE via the F2 key from the INPUT SCORES, UTILITIES, and WEIGHTS menus at the appropriate nodes.

Both files SAMPLE and ZERO contain descriptions of the attributes as the initial rationale at bottom-level nodes. The initial rationale for node 3 4 is shown below. This is called up by hitting the F2 key after INPUT SCORE for node 3 4 is selected. The user may modify this rationale, for example to explain why scores were assigned. The rationale feature has limited word-processing capability. Most operations are natural ones, e.g., backspace deletes the previous character, del deletes the current character, cursor keys move the cursor, etc. Other useful operations are described in Appendix C.


Edit the scores shown below. Use arrows to move between scores and
'ESC' to end. F2 = rationale edit & review, F9 = calculate utility
3 4 Overall->Service->RunTime
Rationale Editing and Review Mode
Run time is the amount of time required to run the program with a
realistic set of input data. This attribute refers only to the time that
the computer program takes to run; the time needed for the user is under
PERFORMANCE factors.

APPENDIX A: OVERVIEW OF MULTIATTRIBUTE UTILITY (MAU)

Multiattribute utility (MAU) analysis (Keeney and Raiffa, 1976) is a methodology that is grounded in the mathematics of measurement. MAU provides an appropriate procedure for evaluation in cases where multiple objectives are important. MAU models reflect explicitly the relative importance of various performance levels on different objectives and the tradeoffs among objectives.

The key stages in a multiattribute utility (MAU) analysis, as they relate to testing, are as follows:

- identification of what is to be evaluated (e.g., a particular expert system);
- definition of the criteria, factors, or attributes of value;
- evaluation, or "scoring" of systems against the attributes;
- prioritization of the attributes of value;
- evaluation of systems;
- sensitivity analyses.

IDENTIFICATION OF WHAT IS TO BE EVALUATED

The first step is to determine what is being evaluated. In the case of a system being tested, that system is to be evaluated. It is also important to identify more precisely whether the evaluation includes a particular hardware/software system, the system and its human operators, or something else. This choice will influence the choice of attributes and the scope of testing. For purposes of this illustration, we will assume that a single, well-defined hardware/software system is being tested and that these test results are being evaluated to determine the acceptability of the system to perform functions to assist human operators, but that the operators themselves are not being tested. We further assume for illustration that it is desirable for the test to identify areas of strength and weakness in the system as well as indicate its acceptability.

To use MAU analysis for test evaluation, it is also useful to construct additional hypothetical "benchmark systems" to use as points of reference. For purposes of this analysis, we will specify the following "systems":

- the test system (abbreviated "tes" in TESTER_C), which is the system being subjected to testing;
- a goal system ("pas"), which is a hypothetical system that fully attains every goal on every attribute;
- a failing system ("fal"), which is a hypothetical system that fails on every attribute;
- a marginal system ("mar"), which is a hypothetical system that, on balance would just manage to pass the test, considering its performance over all attributes.

Introduction of these hypothetical systems enables a tester to apply the test criteria on a consistent, comparative basis, and to highlight areas of deficient and superlative performance. Of the hypothetical systems, the marginal one may be most difficult but most important to describe. Any given system under test is likely to have some areas where it falls short of goals and others where it exceeds goals. In addition, some of the goals may be set as ideals that could not be expected to be met. The marginal system provides a way for the tester to interpret performance in a way that recognizes these possibilities, and to specify *in advance* a minimal level of acceptable performance. This specification in advance removes some of the subjectiveness of the process by setting an overall level of acceptability before test results are known. Note that the marginal system will not generally be unique. Many possible combinations of performance against attributes may be minimally acceptable. However, when the MAU model is fully specified, all of these marginal systems should receive about the same overall evaluation. Specification of one of these systems thus aids in the overall evaluation of the actual system being tested.

IDENTIFYING ATTRIBUTES OF VALUE

Figure 1 shows the attributes of value developed in *Volume 1: Handbook for Testing Expert Systems* as incorporated into TESTER_C. Notice that each node in the attribute hierarchy is labeled with an outline code. To use this

outline code with TESTER_C, put a space between numbers, e.g., 1 1 1 1 Redundant Rules. This is usually a space between digits; the exception is with the longer numbers as under "Service," e.g., 3 11 Formats (*not* 3 1 1). Notice that TESTER_C includes several nodes labeled "OTHER;" these may be specified, as needed, by the tester.

EVALUATION ON ATTRIBUTES

Next, a scale is developed for each bottom-level attribute that relates improvements on the scale to the value to the organization. Often this scale can be developed using natural standard units (e.g., minutes for time, percent correct for accuracy) when such units exist. The relationship between changes on the scale and the value of the changes is then established, and the value is transformed for modeling purposes into a standard scale, such as a 0 to 100 scale. In cases where no natural units exist, a relative value scale, such as a 0 to 100 point scale, could be used directly. Here, it is important to define the points on the scale carefully in terms of the attribute being represented so that unbiased assessments can be made.

For purposes of this report, we define the scales of utility such that a 0 represents failure on that attribute and 50 represents meeting fully the performance goal on the attribute. This choice is arbitrary in the sense that these levels of performance could be assigned any numbers, for example, 0 and 100, 0 and 1000, or 27 and 78. However, the points are not arbitrary in their meaning; 0 is assigned consistently to the failure level, and 50 is assigned consistently to the level of full satisfaction. This assignment provides a basis for consistent interpretation of the analysis and provides the kind of consistency that reduces bias from the assessments. The scale also allows value to be attached to performance that exceeds the goal, by scores greater than 50. Scales represent ratio judgments of value in the following manner. A score of 25 is halfway (in value) between failure and full goal attainment. This provides for convenient and consistent interpretation of scores. However, scores represent value on individual attributes only, and a score on one scale is not generally comparable to a score on another attribute. The weighting procedure described below provides a means for comparing across attributes.

Since a later step in the analysis makes comparisons across attributes, the method implies that attributes are, to some degree, compensatory. That is, a low score on some attribute can be compensated for, at least partially, by high scores on other attributes. If such is not the case, TESTER_C allows for the establishment of thresholds on individual attributes. A threshold is a minimal level of performance on a single attribute that must be met. Failure to meet this performance renders the system under test unacceptable regardless of its performance in other areas.

Suppose the goal on set-up time were five minutes and that sixty minutes of set-up time were considered totally unacceptable. Suppose, further, that the shorter the set-up time, the better, with instantaneous set-up ideal and that an increase from 5 to 15 minutes was considered as serious as an increase from 15 to 60 minutes. This would result in the utility curve shown in Figure A-1.

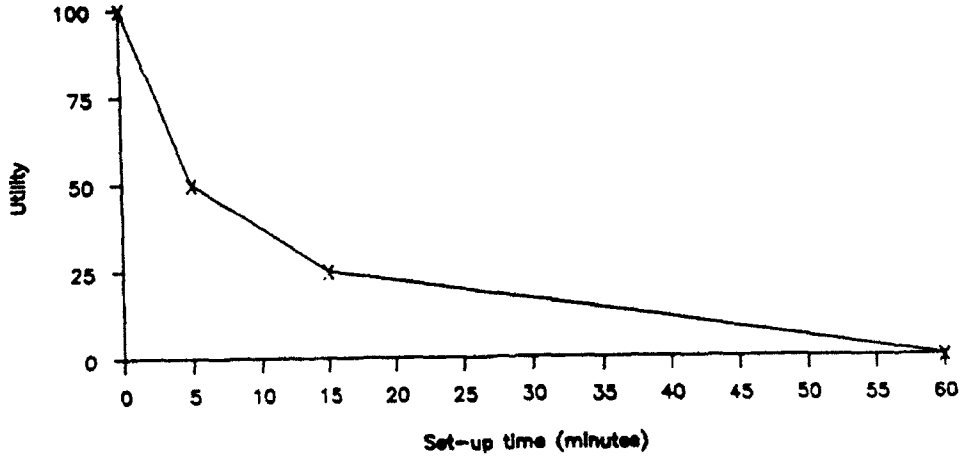


Figure A-1: Utility Curve for Set-Up Time

A utility curve reflects value and thus is inherently subjective. However, the explicitness with which these values are used in the MAU analysis removes the possibility of evaluation on the basis of a hidden agenda. The results and outcome of an MAU analysis are reproducible by people who share the same judgment over appropriate values and tradeoffs to use, and differences in evaluations by different people can be traced to specific

differences in their value structures, which are open to inspection in the MAU analysis.

The utility curve could take on many different shapes. In some cases, utility increases slightly until some point is reached and then it rises dramatically. In other cases, utility is "all or nothing;" that is, no value is perceived until a certain point is reached, then all value is obtained. It is also possible for utility to rise up to a target point and then drop off (e.g., for bias, which runs from -1 to + 1, with a target of 0). These situations could lead to the following types of curve:

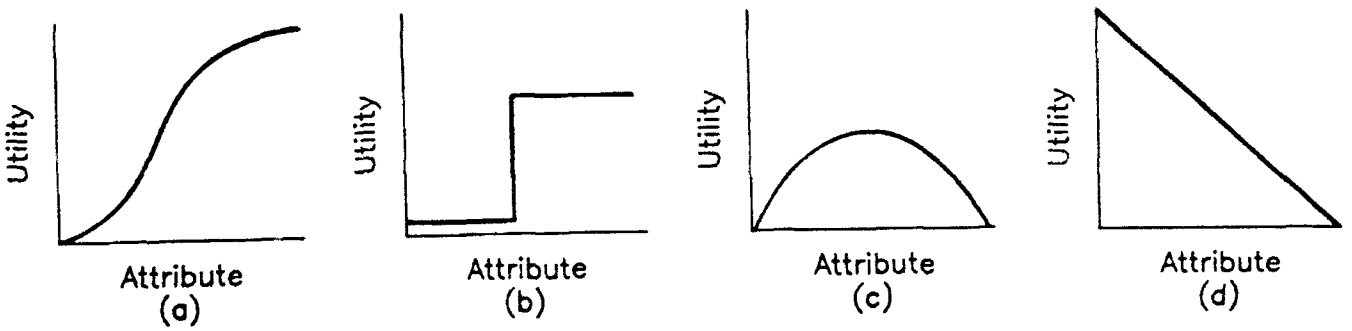


Figure A-2: Some Possible Shapes for Utility Curves

There is also no requirement that utility curves be continuous. Sometimes the attribute can be measured in discrete terms, or categories, even though there is a continuous range for the measure. An example is shown in Figure A-3.

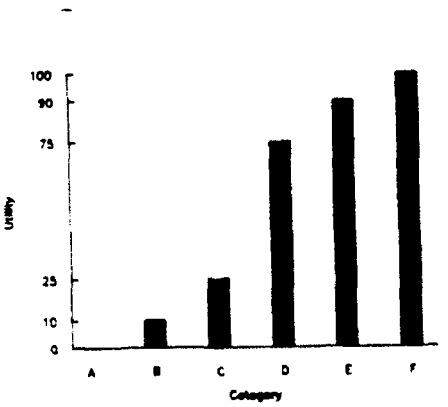


Figure A-3: Utility Curve for a Discrete Categorical Variable

Two important features of utility assignment are worth noting. First, the horizontal axis for each factor is determined uniquely for that factor. Common sense and logic dictate the appropriate measure. Second, it is not necessary to develop formally the utility curves themselves. Once the logic behind the curves is apparent, scores might be directly assessed.

Often, there is not a readily quantifiable measure for an attribute. In these cases, verbal descriptions of relative measure can be used. An example is the categorical attribute shown in Figure A-3. Scaling terms such as High/Medium/Low, Yes/No, Poor/Good/Very Good, and Go/No might also be used to define measurement scales.

PRIORITIZATION OF THE ATTRIBUTES (WEIGHTING)

In the scoring systems described above, an evaluation scale from 0 to 100 was developed for each factor. However, each scale is defined independently of all others, and the resulting scores are not directly comparable. In any real test, some attributes carry more importance than others, and a measure of the priority, or relative importance, of each factor is necessary for an overall evaluation. This is accomplished through a weighting system. As with the scores, weights are judgments, and could vary from organization to organization or from tester to tester. MAU analysis makes such weights explicit, however, and available for review. (Weights should also be subjected to sensitivity analyses as discussed below.)

The most common perception of a weight is that it answers the question, "How *important* is attribute A relative to attribute B?" Unfortunately, such a question often obscures the issue of evaluation. A more pertinent question to ask is, "How important is the *difference* along the range in values for attribute A versus the *difference* for attribute B?" The subtle differences in wording of these two questions is extremely important. The latter question includes both the importance of the attribute as well as the "swing" in the range of values on the attributes. The interpretation of weights and the procedure for assessing them depends on the form of the model to be employed to aggregate the single-attribute scales. The theoretical basis for a variety of aggregation models has been developed (see Keeney and Raiffa, 1976), but an

additive aggregation rule, which is used in TESTER_C, is appropriate or a sufficient approximation in most cases. (The additive rule is appropriate if "additive independence" conditions are met.)

Weighting can be accomplished top-down or bottom-up. Top-down weighting is usually easier. In the top-down approach, the analyst begins at the highest-level node in the hierarchy and assesses the relative differences among attributes. A common approach assigns a weight of 100 to the most important swing. Other weights are then assigned using ratio judgments—that is, if the swing on an attribute is judged to be twice as important as the swing on another attribute, the former would carry twice the weight of the latter. Furthermore, with additive independence, weights can be compared in an additive sense. If attribute A is weighted at 100, attribute B at 75, and attribute C at 50, this implies that, for example, the combined effect of 50-point swings in both B and C (added weights equal 125) is more important than a 50-point swing on A (weight of 100). Such comparisons can serve as a good calibration check on the weights.

For consistency in the analysis, the weights are often normalized to sum to 1.00 by adding the assigned weights and dividing each by the sum. This is done automatically by TESTER_C. The basis for assigning weights might be in a statement of requirements or other guidance provided to the tester, and this could well vary from one test to another. See Chapter 7 of *Volume 1: Handbook for Testing Expert Systems* for more information. In fact, some guidance may be such that some of the attributes are irrelevant. If this is the case, a weight of zero could be assigned to the attribute. In any case, assigning weight *before* the test is conducted is a recommended procedure to reduce bias.

EVALUATION

After scores have been assessed against the attributes and weights have been assigned, evaluations can be determined. Since an additive MAU analysis is used in TESTER_C, the overall evaluation of an alternative is a weighted average of assessed scores, with the exception that a system is regarded as a

failure if any score falls below a threshold on any attribute. (This is described in more detail in *Volume 1: Handbook on Testing Expert Systems*.)

Although the numerical results of a MAU analysis provide a compact representation of the evaluation, they are not the only results of the analysis. The numerical output can also direct the tester to areas of strength and weakness in the system under test and thus provide the basis for suggested improvements. The numerical output also summarizes explicitly the judgments used and thus provides a basis for building a verbal case for or against the system. Also, the explicit numerical representations of judgment, especially in the form of weights, provide a means of identifying important differences of opinion if differences exist between testers and evaluators.

SENSITIVITY ANALYSIS

Several reasons recommend sensitivity analyses for most test evaluations. First, some parts of the analysis may not be known with a high degree of accuracy for any of a number of reasons. While it is desirable to design and conduct tests that provide highly accurate assessments, lack of resources or other reasons sometimes prevent the level of accuracy desired. At this point, the test evaluator may decide to include or exclude the data from consideration in the evaluation. We recommend including the data but running sensitivity analyses over the range of uncertainty. Use of judgmental information is another reason to perform sensitivity analyses.

Three major types of sensitivity analyses are supported by TESTER_C. First, the scores that have been assessed can be modified to determine if results change. This type of sensitivity analysis is appropriate in cases where scores were assessed with inadequate test data or where judgmental assessments were made. Generally, results are reasonably insensitive to minor changes in scores, especially with an analysis that uses the whole MAU framework. Next, several weights can be changed and the overall scores recalculated. This is useful in examining large-scale changes to the model (such as using weights for a different evaluator), but does not make it easy to isolate causes of change or disagreement. These first two types of sensitivity analysis are performed using INPUT options in TESTER_C. A third

sensitivity analysis is to vary one weight at a time and identify the regions where evaluations change. Typically, one factor is chosen and its weight is allowed to vary over a wide range. As the weight increases, the total weight of the other factors must decease but the weights are kept in the same relative proportion to each other. This third sensitivity analysis is supported under the SENSITIVITY part of TESTER_C.

REFERENCES

Hayes, P.J. (1981). "The Logic of Frames." B.L. Webber and N.J. Nilsson (Eds.), *Readings in Artificial Intelligence*. Palo Alto, CA: Tioga, 451-458.

Keeney, R.L. and H. Raiffa (1976). *Decisions with Multiple Objectives*. New York: Wiley.

APPENDIX B: ATTRIBUTE DEFINITIONS AND SUGGESTED SCALES

ATTRIBUTE DEFINITIONS

Figure 1 shows our MAU proposed framework for testing and evaluating expert systems. The overall assessment of the expert system is composed of five criteria: knowledge base, inference engine, service requirements, performance, and usability. These are subdivided to the level of attributes as described below.

KNOWLEDGE BASE. These attributes refer to the structure and content of the expert system's knowledge base. While the descriptions below are phrased in terms of a rule base, analogous attributes would apply to a frame-based expert system. (See Hayes, 1981, for a discussion of the logical equivalents of rule-based and frame-based systems.)

Structure

Logical Consistency. The following attributes would limit the consistency (or correspondence) and efficiency of a knowledge base. Redundant rules are rules or groups of rules that have essentially the same conditions and conclusions. Redundancy can be due to duplicate rules or the creation of equivalent rules (rule groups) by wording variations in the names given to variables, or the order in which they are processed. Subsumed rules occur when one rule's (or group of rules') meaning is already expressed in another rule (or group of rules) that reaches the same conclusion from similar but less restrictive conditions. Conflicting rules are rules (or groups of rules) that use the same conditions, but result in different conclusions, or rules whose combination violates principles of logic (e.g., transitivity). Circular rules are rules that lead one back to an initial (or intermediate) condition instead of a conclusion.

Logical Completeness. A knowledge base is complete if it has no holes or gaps in its logic. The following attributes indicate a logical incompleteness. Unreferenced attribute values are values on a condition that are not defined; consequently, their occurrence cannot result in a conclusion.

Illegal attribute values are values on a condition that are outside the acceptable set or range of values for that condition. An unreachable conclusion is a conclusion that cannot be triggered by the rules combining conditions. Dead ends are rules that do not connect input conditions with output conclusions.

Content

Functional Completeness is the extent to which the knowledge base addresses all domain conditions. All desired inputs: the knowledge base can handle all input conditions that need to be addressed. Application/conclusion completely covered: the knowledge base can trigger all output conclusions that need to be addressed. Identified knowledge limitations: the rules in the knowledge base can tell the user if input conditions currently being processed cannot be addressed. Analogously, if the expert system is such that a user can specify a conclusion in order to identify the input conditions that would generate it (e.g., as in a backward-chaining system), an expert system that was knowledgeable of its limitations would tell users if a conclusion currently being processed as input could not be addressed.

Predictive Accuracy. The following attributes address the accuracy and adequacy of the knowledge base. Problems here may also be related to problems of performance. Accuracy of facts: the quality of the unconditional statements in the knowledge base. Accuracy of rules: the quality of the conditional statements in the knowledge base representing expert judgment. Knowledge representation acceptability: whether or not the scheme for representing knowledge is acceptable to other domain experts and knowledge engineers. Adequacy of source: the quality of the persons or documentation used to create the knowledge base. Modifiability of knowledge base: the extent to which the knowledge base can be changed and the control over that change.

INFERENCE ENGINE: the extent to which the inference engine provides error-free propagation of rules, frames, probabilities, or other representation of knowledge or uncertainties used in the system.

"SERVICE" refers to aspects of the system (computer and others) in which the expert will operate.

Computer System. Design: the extent to which the expert system runs on the approved computer hardware and operating system and utilizes the preferred complement of equipment and features. In some cases, the design system will be stated in a requirements document; in other cases, the tester may need to survey available equipment at the intended installation. Portability: how easily the expert system can be transferred to other computer systems.

Computer Usage. Set-up time: the amount of time required for the computer operator to locate and load the program (if any) and the time to activate the program. Set-up time should be measured under the expected operating conditions. Run time: the amount of time required to run the program with a realistic set of input data. This attribute refers only to the time that the computer program takes to run; the time needed for the user is under PERFORMANCE factors. Space requirements: the amount of RAM, disk, or other space required by the program. Hardware reliability: the percentage of time the computer system could be expected to be operating effectively. Hardware capability: the computer system's total amount of RAM, disk, or other space. Effect of feature use/jumping: the extent to which moving from various parts of the program causes errors. Degradation: how well the program saves data and analyses and permits continuation after an unexpected program or system crash or power outage. Handling input errors: the extent to which the program prohibits a program crash and tells the user what to do after an input mistake.

System Integration. Formats: the extent to which the program uses input and output formats that are consistent with the intended use. This includes any mandated or standard formats that are specific to the intended user organization. Data requirements: the extent to which the program's data requirements are consistent in content, quantity, quality, and timeliness with those available to the intended user organization. The expert system should also be able to interact with specified and appropriate databases and communications systems. Documentation: the adequacy of material regarding the program's use and maintenance. Copies of computer code and its supporting

documentation should be complete and understandable, and should allow maintenance by the user organization. (All applicable software documentation standards should be met.) Skill requirements: the extent to which the program can be operated by appropriately skilled individuals. The appropriate skill requirement includes grade level (for military enlisted, military officer, or civilian personnel), users' technical background, and training requirements. The appropriate level may be specified in requirements or may be determined by reference to the organizational setting of its intended use and to the personnel assigned to that setting.

PERFORMANCE refers to the operation of the expert system and the user. It includes both comparisons with ground truth and judgmental assessments.

Performance against Ground Truth. Speed: the amount of time it takes a user working with the expert system to solve representative problems. Accuracy: the degree of overlap in the distributions of belief values when the hypothesis is true versus false (see Chapter 5 of Volume 1, *Handbook for Testing Expert Systems*). Bias: the difference in the proportion of false negatives (hypothesis is true but system says false) to false positives (hypothesis is false, but system say it's true) (see Chapter 5 of Volume 1, *Handbook for Testing Expert Systems*).

Judgmental Performance. Response time: the judgments of users regarding the adequacy of the amount of time the expert system takes to react to inputs. Time to accomplish task: the judgments of users regarding the adequacy of the amount of time required to perform the task when using the expert system. Quality of answers: the judgments of users and experts regarding the system's capability. Quality of reasons: the judgments of users and experts regarding the adequacy of the system's justification for its answers.

USABILITY is the extent to which the expert system, or parts of the expert system, is used, is acceptable to individuals, and is acceptable to the organization.

Observable Usability includes aspects of usability that a tester can observe (or a system can record) during a test without asking the test sub-

jects. Extent of use: how much users employ the expert system to perform the task (e.g., the proportion of time that the system was used to accomplish tasks assigned in a test). Manner of use: the way in which users employ the system and its features, including the procedures to access different modules, the way that intermediate and final outputs are incorporated into the user's results, and the use of interfaces. Features used: the extent to which different aspects of the expert system are employed by users.

Opinions about Usability. Confidence: how confident users feel in taking actions based on working with the expert system. Ease of use: how easy users judge the system is to use after they have completed training and become familiar with the system. Acceptability of person/machine interaction process: the extent to which users assess that they and the system are performing the tasks or activities for which they are best suited. Acceptability of results: the users' judgments regarding the adequacy of the system's capability. Acceptability of representation scheme: the users' judgments regarding the adequacy of the system's way of presenting knowledge. Input/output: the user's judgment about the adequacy of the extent, display, and manner of accessing the expert system's input and output.

Scope of Application: the users' judgments regarding the adequacy of the expert system in addressing domain problems.

Explanation. Adequacy of presentation and trace: the users' judgments regarding the acceptability of the system's presentation of its reasoning process. Transparency of expert system: the extent to which the system's reasoning process is clear and understandable to its users.

Organizational Impact. Impact on work style, workload, skills, and training: the judgments of users regarding the impact of the expert system on how they do their job, or the skills and training required to perform it effectively. Impact on organizational procedures and structure: the judgments of users regarding the impact of the expert system on the organization's operations.

SUGGESTED SCALES FOR ATTRIBUTES

Appropriate scales for the attributes may differ from one expert system to another. Although it is impossible to set scales that will apply to every expert system in every operating condition and every intended use, we can suggest scales that the tester should consider. These are given below. Some suggested scales are simple "Yes or No" categorizations, others are natural units such as minutes, still others are percentages. These may be helpful in establishing consistent frames for assessing the performance of a system that is being tested. We have avoided guidance on specific criteria of acceptability (e.g., "set-up time should be less than 10 minutes") because such criteria depend critically on specifics of the expert system and its intended use. We feel that generalizations of this nature would not be supportable. In general, these scales should be set before a test is begun. In addition, the relationship between performance on the scales and the utility of that performance should also be established, for example as discussed in Appendix A.

KNOWLEDGE BASE

- *Logical Consistency*:
  - *Redundant Rules*. Suggested scale: Percentages. The tester will examine the rule base and determine the percentage of individual rules and rule sets that are redundant. The tester may be able to perform a manual walk-through of small rule bases, but use of multiple software testers is better because the tedious nature of the task will no doubt result in errors. If an automated "static tester" were not available for a large rule base, some sampling procedure would be required.
  - *Subsumed Rules*. Suggested scale: Percentages. Same rationale as that presented for "redundant rules."
  - *Conflicting Rules*. Suggested scale: Number. Our definition was that conflicting rules used the same (or very similar) initial conditions, but resulted in either different conclusions, or violations in logic. In contrast to redundant or subsumed rules, which affect system efficiency, conflicting rules could well result in bringing the system to a halt unless there is an effective conflict resolution mechanism; at the least, it results in a logic error. Unless the initial conditions for conflicting rules are extremely rare, even 1 or 2 conflicting rules (or rule

sets) that essentially crash the system may be unacceptable even though their percentage in the rule base may be extremely small.

- *Circular Rules*. Suggested scale: Number. Same rationale as for "Conflicting Rules."

- *Logical Completeness*.

  - *Unreferenced Attribute Values*. Suggested scale: Number, because the effect is on system effectiveness, not efficiency. (This assumes that the unreferenced attribute values could occur in the operational environment. If they cannot, then they are more like "Unnecessary If Conditions," affecting the efficiency with which the system examines the rule base.)

  - *Illegal Attribute Values*. Suggested scale: Number. Same rationale as for "Unreferenced Attribute Values."

  - *Unreachable Conclusion*. Suggested scale: Number. Same rationale as for "Unreferenced Attribute Values."

  - *Dead Ends*. Suggested scale: Number. [Note: Effectiveness vs. efficiency concern.]

- *Functional Completeness*.

  - *All Desired Inputs*. Suggested scale: Number. This again addresses effectiveness. It should be remembered that this "Functional Completeness" assessment is made by reference to a requirements statement, or, if that does not exist, by domain experts. Consequently, each violation on this attribute may need to be examined because even one or two input omissions may have a significant impact on the utility of the system. The tester should consider placing a threshold of "no omissions" on this attribute.

  - *Application/Conclusion Completely Covered*. Suggested scale: Number. Same rationale as for "All Desired Inputs."

  - *Identified Knowledge Limitations*. Suggested scale: Yes or No. Most likely, the expert system either claims to have this capability or it does not, and the feature either works or it does not.

- *Predictive Accuracy*

  - *Accuracy of Facts*. Suggested scale: Number. Each "inaccurate fact" needs examination in order to assess the utility score on this attribute. Accuracy should be determined by reference to an acknowledged source.

  - *Accuracy of Rules*. Suggested scale: Number. This can usually be determined only by an expert or, preferably, by a group of

experts. Each "inaccurate rule" needs to be examined to assess utility score on this attribute.

- *Knowledge Representation Acceptability.* Suggested scale: Yes or No. The implemented knowledge representation scheme is acceptable or not to other domain experts and knowledge engineers. The tester may want to get the opinions of several knowledge engineers and domain experts, if possible, for this assessment. "Other" knowledge engineers might conclude, on either effectiveness or efficiency grounds, that (a) an inappropriate representation scheme was used, or (b) that an appropriate scheme was not implemented well. Such assessments may be particularly important when the expert system is in the prototype stage.

- *Adequacy of the Source.* Suggested scale: Yes or No. It is possible for a source to provide accurate information, but for it to be so limited as to be inadequate. This attribute will most likely require the opinions of a domain expert or panel of experts.

- *Modifiability of Knowledge Base.*

  -- *Control Over.* Suggested scale: Yes or No. A software tester can assess whether accessibility to the knowledge base is controlled or not. A requirements statement, sponsoring agency, users, and perhaps security analysts and domain experts may be needed to assess whether the level of control is acceptable or not.

  -- *Expandability* (by human/machine). Suggested scale: Yes or No. Again, a tester can assess whether the knowledge base can be increased (i.e., expanded), decreased or, in general, modified by humans and, perhaps most interestingly, by machines. A requirements statement (or the system's sponsoring agency) may provide an assessment of whether such expandability is desirable. Domain experts working with AI specialists would probably be required to assess whether the changes were acceptable. [Note: Acceptability, in terms of performance, could be determined by statistical analysis of test cases where subjects changed the knowledge base.]

"SERVICE"

- *Computer System.*

  - *Design.* Suggested scale: Yes or No. Consistent with the definition, the expert system either runs on the approved computer hardware and operating system (and utilizes the preferred equipment and features) or it doesn't. If it does, then it passes. If it doesn't, then it fails; the utility score (e.g., between "0" and "50") would depend on the type of incompatibility problems found by the software tester. [Note: If the system scores "0" on

"Design," which means that it does not run on the approved hardware and operating system, then its values for "Set-Up Time," "Run Time," "Space Requirements," etc. are all tied to the hardware the system does run on. For an early prototype, this may be quite acceptable, for "Design" may have a low weight. However, in the later stages of development, there may be a noncompensatory threshold rule where a "0" on "Design" results in an unacceptable score overall.]

- *Portability*. Suggested scale: Yes or No for comparable machines. For example, if the expert system was developed for an IBM AT, then it would "pass" on portability if it could run on AT-compatibles of similar power. If it could also run on an IBM PC (or compatibles), then it would get a utility score greater than "50," depending on whether it ran with all its features. If it couldn't run well on an AT-compatible, it would receive a score less than "50." If it couldn't run at all on an AT-compatible (or a PC), it would get a score of "0." The same logic holds for mainframes, and for going between mainframes and personal computers. [Note: It is possible that the system is portable with one type of hardware, but not another. The tester should refer to any statement of requirements to determine the range desired. The hardware "types" would receive weights to obtain a total score.]

- *Computer Usage.*

  - *Set-Up Time*. Suggested scale: Minutes. The software tester may want to calculate the average and standard deviation for this attribute. However, that requires that the software tester perform the set-up a number of times (e.g., 10). The amount of time required for such repetition, particularly for measuring other attributes (e.g., "Run Time" or "Ground Truth Performance") is probably unacceptable unless the attribute is very important.

  - *Run Time*. Suggested scale: Minutes. The tester should record this for all test cases (to the extent possible) and may use statistical summaries (e.g., mean and standard deviation) in the assessment.

  - *Space Requirements*. Suggested scale: The amount of RAM and disk space required to run the system. Standards of acceptable size may be stated in a requirements document. Otherwise, acceptable sizes might be determined by the tester based on the total available.

  - *Reliability (Hardware)*. Suggested scale: Percentage of time in a 24-hour day (or during specified periods) that the computer (i.e., hardware) is operating effectively. [Note: The tester might want to expand the definition to include software if the expert system requires distributed databases that require periodic updating and possible "down time," independent of the hardware.]

  - *Capability (Hardware)*. Suggested scale: The computer system's total amount of RAM and disk space. The importance of this will

be related to how close the expert system comes to using all available space.

- *Feature Use/Jumping*. Suggested scale: Number (and type). Each case where moving from one part of the program to another caused an error would have to be examined because of its potential effect on system effectiveness.

- *Degradation (Graceful?)*. Suggested scale: Number (and type). The concern is on the effect of ungraceful degradation on effectiveness. In some operational environments, even one ungraceful degradation would be unacceptable. This attribute might also be measured on a "Yes or No" scale on the assumption that the system should degradate gracefully, regardless of the cause precipitating the system crash or power outage.

- *Handling Input/Output Errors*. Suggested scale: Number (and type). Same rationale as for "Degradation (Graceful?)." This could also be "Yes or No" on the assumption that the system could (or couldn't) tell the user what to do after an input mistake, but it's possible that this capability could exist in some modules and not others.

• *System Integration*

- *Formats*. Suggested scale: Number (and type). Identify all inconsistencies with input and output formats specified in the requirements document or other appropriate source.

- *Data Requirements*. Suggested scale: Number (and type). Identify all inconsistencies in the content, quantity, quality, and timeliness of the system's data requirements and those specified in the requirements document or other appropriate sources.

- *Documentation*. Suggested scale: Acceptable or Unacceptable. All applicable DoD software documentation standards were met. Standards that were failed should be identified by the software tester. If DoD standards aren't appropriate, the software tester should assess whether the expert system's documentation is, in general, complete and understood or not. Problem areas need to be identified. This assessment should be separately performed for (a) the user's manual, (b) the operator's manual, and (c) the computer code and its supporting documentation.

- *Skill Requirements*. Suggested scale: Yes or No. This may be difficult to assess. The concern is whether, prior to giving the system to users, software testers could make an initial assessment of whether targeted users have the required background skill to effectively operate the system. After examining the (1) requirements document and (2) documentation describing the users' organizational setting, this may be an easy or difficult assessment. The binary "Yes/No" measurement scale is a conservative scale. That is, passing the "Skill Requirements" attribute should be easy to assess or the system fails. For example, for

one Army expert system, this proved to be a critical issue. The terminology used in the system was the terminology of the experts and proved beyond the entry level of the user actually causing the users to interact with the system in an incorrect manner. This was partially because the skill level of the users was based on completion of a certain course which no longer contained many aspects that were in the course when the experts took the course.

PERFORMANCE

- *Ground Truth.*

  - *Speed.* Suggested scale: Minutes. Consistent with the previous discussions, software testers should calculate the mean and variance for the amount of time it takes the (test) users to solve (representative) problem scenarios working with the expert system.

  - *Accuracy (d*).* Suggested scale: Probability that two points, one taken from the Positive distribution (i.e., the hypothesis is true) and one taken from the Negative distribution (i.e., the hypothesis is false) will be in reverse order. That is, the probability that the belief value of a point x_p from the P distribution is lower than the value of a point x_n from the N distribution:

$$d^* = p(x_p < x_n | x_p \epsilon P,\ x_n \epsilon N).$$

    (See Chapter 5 of Volume 1, *Handbook for Testing Expert Systems*, for details.)

  - Bias (B*). Is calculated by the following formula:

$$B^* = \frac{\text{\# false alarms}}{\text{\# in } S_N} - \frac{\text{\# false positives}}{\text{\# in } S_p}.$$

    (See Chapter 5 of Volume 1, *Handbook for Testing Expert Systems* for details.)

USABILITY

- *Observable.*

  - *Extent of Use.* Suggested scale: Proportion of time the system was used for task accomplishment. Again, propose calculation of the mean and variance for this distribution.

  - *Manner of Use.* Suggested scale: Type and Percentages. The software tester identifies the different ways in which users employed the expert system and its features. Then the tester

calculates the percentage of users who used the system in each of the identified ways.

- *Features Used*. Suggested scale: Percentages. Tester calculates the percentage of users who used each of the system's basic features when solving the problem scenario.

JUDGMENTAL PERFORMANCE AND THE REST OF USABILITY

Two forms of questionnaires are provided for these attributes in the Appendix of Volume 1, *Handbook for Testing Expert Systems*. These questionnaires should be used with a sample of test subjects and the means and variances calculated for assessing performance on the attribute.

APPENDIX C: RATIONALE

The software allows full text editing on the rationale. It supports the following keys:

Esc	Quit editing.
Gray -	Cut the marked region into a buffer.
Gray +	Copy the marked region into a buffer.
Del	Delete the marked area if there is one, else delete the character under the cursor.
Ins	Paste the cut buffer into the text.
Alt-M	Mark region by rows.
Alt-C	Mark region by columns.
Alt-S	Search for a string.
Alt-R	Search for a string and replace it.
↑	Moves the cursor up one character.
↓	Moves the cursor down one character.
→	Moves the cursor right one character.
←	Moves the cursor left one character.
Home	Moves to beginning of the current line.
End	Moves to the end of the current line.
PgUp	Moves up one page.
PgDn	Moves down page.
Ctrl-PgUp	Goes to the beginning of the text.
Ctrl-PgDn	Goes to the end of the text.
←	Delete the character to the left of cursor.
Tab	Inserts a tab character.

APPENDIX D: SBIR RIGHTS NOTICE (JUN 1987)

These SBIR data are furnished with SBIR rights under contract No. DAEA18-88-C-0028. For a period of 2 years after acceptance of all items to be delivered under this contract, the Government agrees to use these data for Government purposes only, and they shall not be disclosed outside the Government (including disclosure for procurement purposes) during such period without permission of the contractor, except that, subject to the foregoing use and disclosure prohibitions, such data may be disclosed for use by support Contractors. After the aforesaid 2-year period, the Government has a royalty-free license to use, and to authorize others to use on its behalf, these data for Government purposes, but is relieved of all disclosure prohibitions and assumes no liability for unauthorized use of these data by third parties. This Notice shall be affixed to any reproductions of these data, in whole or in part.